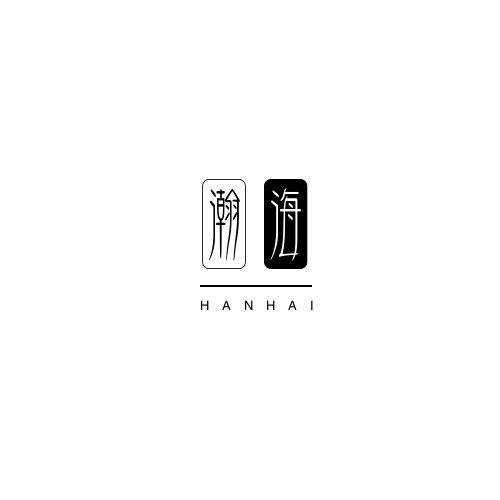


**悠食**



**详细设计**

**团队名称 瀚海开发小组**

****

**团队LOGO**

**姓 名 覃海旭 肖瀚驰**

**学 院 人工智能学院**

**班 级 2020219108**

**学 号 2020212302 2020212304**

**2022年 6 月**

# 1、系统概述

**1.1、系统简介**

悠食是一款致力于在当前社会背景下为用户提供饮食健康管理的微信小程序，是一个独立的项目且全部内容自含**。**

**背景：**当代年轻人因为学习、生活和工作上的压力，缺乏运动，不注重饮食健康习惯，这种现象导致了严重的身体及心理健康问题。基于这种普遍的社会现象，我们决定制作一款基于微信小程序平台开发的饮食管理系统，旨在通过微信小程序这个日常使用频繁的平台来对用户的日常饮食进行管理并给出建议，从而改善用户的健康状态。

**应用目标：**

1. 通过用户的身体条件基本信息计算用户的基本代谢水平

2. 通过用户的日常饮食来计算热量摄入

3. 将用户的日常饮食摄入与用户的基础代谢水平相结合，并进行分析比较，以对用户的日常饮食与运动习惯做出建议

**1.2、术语表**

定义系统或产品中涉及的重要术语，为读者在阅读文档时提供必要的参考信息。

|  |  |  |
| --- | --- | --- |
| **序号** | **术语或缩略语** | **说明性定义** |
| 1 | MET | metabolism,基本代谢水平 |
| 2 | RCI | Recommended calorie intake，推荐摄入热量 |
| 3 | openid | openid,微信小程序用户身份识别码 |
| 4 | CT | calories taken，已摄入热量 |

**1.3、系统运行环境**

**硬件平台**：能运行微信客户端的手机、电脑等设备

**操作系统**：Android、iOS、Windows、macOS等能运行微信客户端的操作系统

**UI设计平台：**墨刀客户端

**编程平台：**微信开发者工具

**数据库系统：**微信云开发数据库

**网络协议：**微信小程序数据传输网络协议，使用TCP协议对数据进行传输和接收

**1.4、开发环境**

**设计工具：** 名称：墨刀

版本:v1.1.9

作用：进行前端设计

**工程工具：** 名称：微信开发者工具

版本：Stable 1.05.2203070

作用：进行程序开发

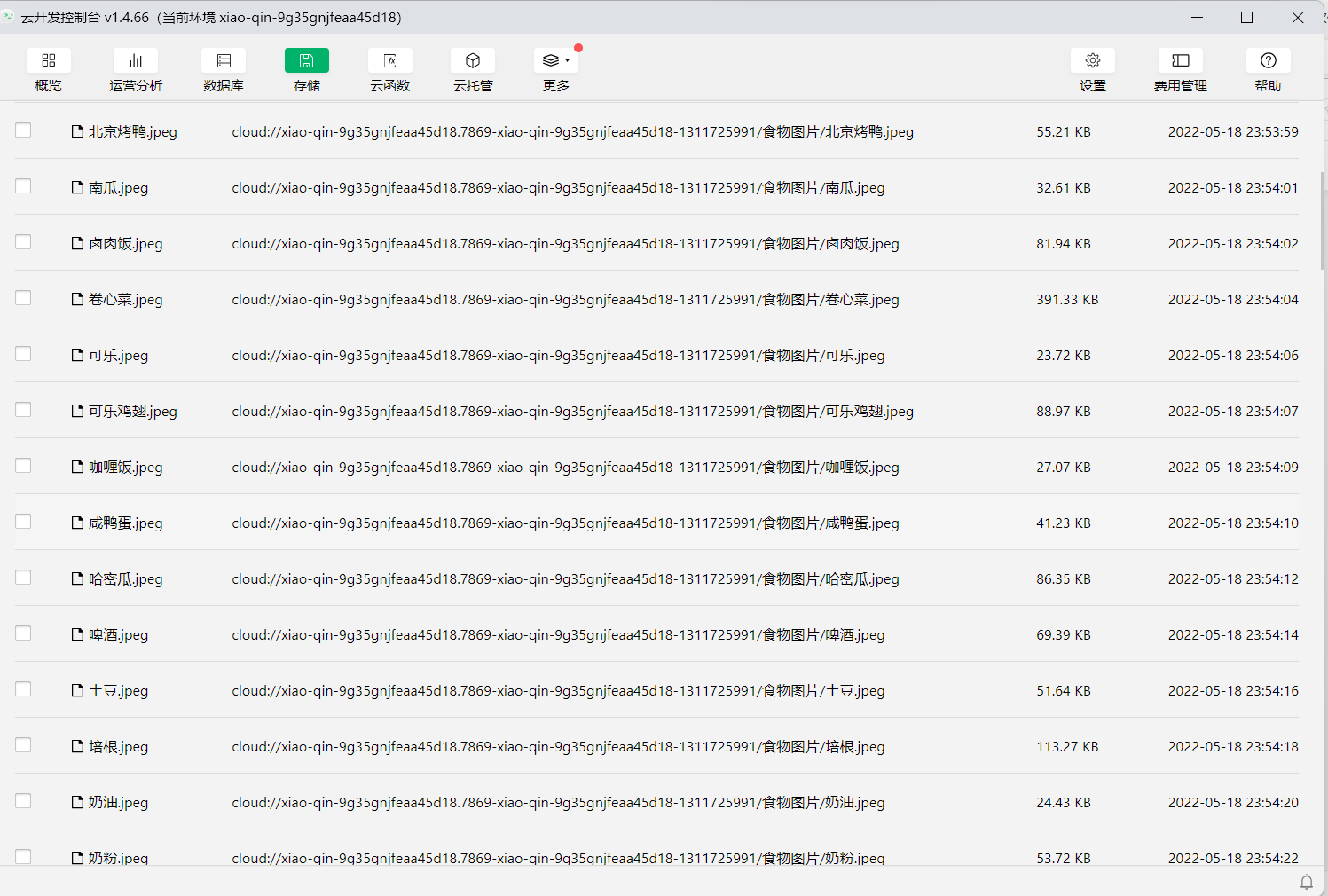
**开发语言**：前端：JavaScript、wxml、wxss

后端：Node.js

# 2、数据结构说明

2.1、常量

在微信云存储中存放着食物图片的信息，数据文件名称为食物图片，其中包含140个食物的图片及链接信息，为小程序前端界面显示食物图片信息提供服务支持



2.2、变量

以下全局变量均创建于app.js文件中，存放在globalData对象中

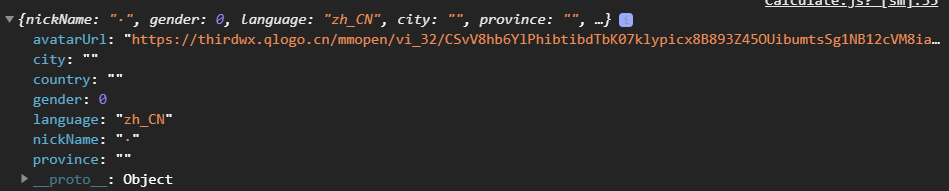


**具体说明：**

**1、userInfo:**类型：对象

功能：存放用户的头像对应url、id等账户信息

说明：键值对存放的数据的具体意义如下图所示

例：

**2、metabolism：**类型：数字

功能：存放用户的推荐摄入热量

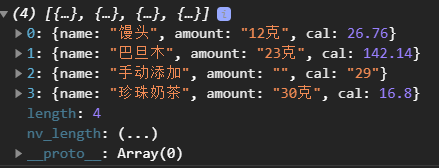
**3、already：**类型：数字

功能：存放用户的已摄入热量

**4、already\_data:**类型：数组

功能：存放用户已添加的食物数据

例：



**5、openid：**类型:字符串

功能：存放用户的身份识别码

**6、date：**类型：字符串

功能：存放从用户手机端（或PC端）获取的日期信息

2.3、数据结构

后端数据库：

**集合food**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识符 | 类型 | 说明 |
| 数据id | \_id | string | 该条数据身份信息（主键） |
| 名称 | name | string | 食物名称 |
| 单位热量 | cal | string | 食物单位热量（每100g） |
| 类别 | class | string | 食物类别 |
| 图片 | img | string | 食物图片链接（云存储链接） |

**集合history**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识符 | 类型 | 说明 |
| 数据id | \_id | string | 该条数据身份信息（主键） |
| openid | \_openid | string | 用户身份识别码 |
| 摄入热量 | cal | string | 用户历史摄入热量 |
| 日期 | date | string | 用户历史热量摄入对应的日期 |

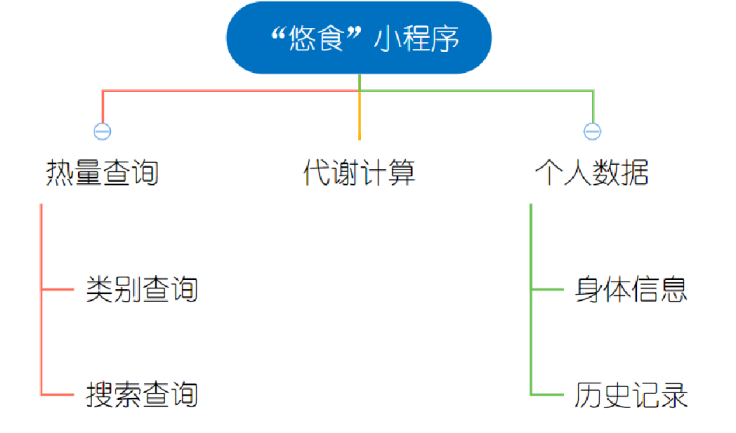
**集合pri\_inf**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识符 | 类型 | 说明 |
| 数据id | \_id | string | 该条数据身份信息（主键） |
| openid | \_openid | string | 用户身份识别码 |
| 身高 | height | string | 用户当前身高 |
| 体重 | weight | string | 用户当前体重 |
| 年龄 | age | string | 用户年龄 |
| 性别 | sex | string | 用户性别 |
| 基本代谢水平 | metabolism | number | 用户基础代谢水平 |

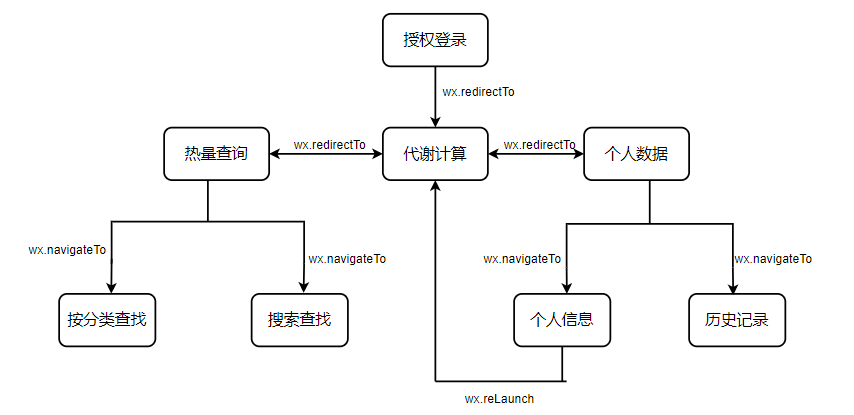
# 3、模块设计

**3.1、软件结构**

**三大模块划分：**

****

**模块间关系：**

****

**3.2、功能设计说明**

我们需要在有限的知识储备、编程能力和时间限制的前提下，开发出具有适用性、有效性、便捷性、可互操作性、可追踪性、复用性、封装性、鲁棒性和满足用户需求的软件产品。

秉承着适用性，为适用于微信小程序平台，需要在首次使用时进行授权登录操作。

秉承着有效性，需要在授权登录后进入代谢计算界面，通过计算得到用户的基础代谢水平后再进行食物热量的计算和管理。

秉承着便捷性，用户可以通过食物分类和食物搜索两种方式对目标食物进行查找。

秉承着可互操作性，对于数据库不包括的食物类型，用户可以手动输入热量来完善饮食摄入信息，并且支持摄入食物克数手动输入。

秉承着可追踪性，用户可以对历史摄入热量的数据进行查询。

秉承着复用性，对于一些可复用的组件(如底部导航栏等),采用引入的方式在不同模块中进行多次复用，使得工程代码简洁、高效。

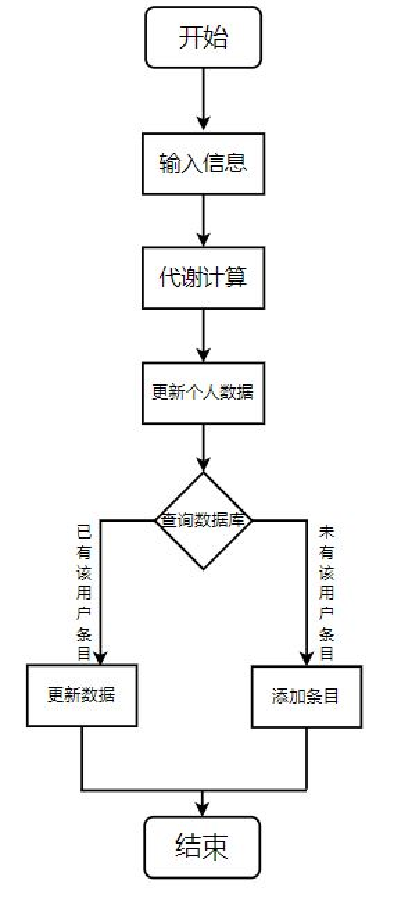
秉承着封装性，不同模块间只通过对应的接口进行调用，而不影响模块内部功能的具体实现。

秉承着鲁棒性，对用户的输入进行限制（如输入身高时调出只含有数字和小数点的键盘）来避免一些非法输入，对于其它类型的非法输入，软件会通过跳出弹窗等方式来进行警告。

**3.3、模块1 代谢计算模块**

详细描述各功能模块的功能、数据结构、具体算法和流程等。

3.3.1、设计图



3.3.2、功能描述

功能1：通过用户的基本身体信息计算基本代谢水平

    输入的数据：身高、体重、性别、年龄

    经过相应公式计算：女性基础代谢率=661+9.6\*体重(kg)+1.72\*身高(cm)-4.7\*年龄

             男性基础代谢率=67+13.73\*体重(kg)+5\*身高(cm)-6.9\*年龄

得到的输出：用户的基本代谢水平

功能2：将用户的身体信息和功能1计算得到的基本代谢水平上传至数据库。若数据库中已有该用户条目，则更新该条目；若没有，则添加相应条目。

3.3.3、输入数据

1、身高：数据类型：数字

获取数据媒体：数字输入框

格式:XX.X

有效性检验规则：1、判断是否输入数据（输入字符串是否为空），若为 空，弹出相应的弹窗警告。

2、正则表达式匹配，判断输入格式是否满足XX.XX，对 应正则表达式为/^\d+(\.\d+)?$/。若不匹配，弹出 相应的弹窗警告。

2、体重：数据类型：数字

   获取数据媒体：数字输入框

   格式:XX.X

  有效性检验规则：1、判断是否输入数据（输入字符串是否为空），若为 空，弹出相应的弹窗警告。

2、正则表达式匹配，判断输入格式是否满足XX.XX，对 应正则表达式为/^\d+(\.\d+)?$/。若不匹配，弹出 相应的弹窗警告。

3、年龄：数据类型：数字

获取数据媒体：数字输入框

   格式：XX

   有效性检验规则：1、判断是否输入数据（输入字符串是否为空），若为 空，弹出相应的弹窗警告。

2、限制用户输入：用户在手机上输入年龄时，弹出的键 盘为只能输入数字（不含小数点）的键盘，确保用户输入 的格式正确。

4、性别：性别：数据类型：字符型

    获取数据媒体：滑动选择框

选项：男，女

有效性检验规则：因为滑动选择只有符合要求的选项，故无需有效性 检验规则

以上输入数据的获取条件均为用户在应用代谢计算模块时，通过键盘自愿主动输入

以上数据均通过代谢计算界面的输入框和滑动选择框获取

3.3.4、输出数据

1、基本代谢水平：数据类型：数字

格式：XX.XX

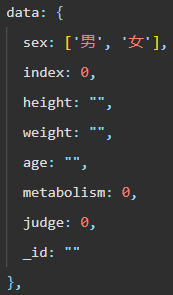
表现形式：弹窗，外加一些文字提示：如“您一天的基础代谢为1800千卡”

2、调试信息：数据类型：字符串

表现形式：在控制台输出相应的提示信息，如“寻找个人信息请求成功”等，用于开发者的调试。

3.3.5、数据设计

本模块的局部变量数据结构如下图所示



**具体说明**：

**1、sex:**类型：数组

功能：存放性别选择框的两种选项（男、女，均为字符串）

**2、index:**类型：数字

功能：记录滑动选择框中选择sex数组中对应的数据索引

**3、height**:类型：字符串

功能：存放用户输入的身高数据

**4、weight:**类型：字符串

功能：存放用户输入的体重数据

**5、age:**类型：字符串

功能：存放用户输入的年龄数据

**6、metabolism:**类型：数字

功能：存放通过计算得到的基本代谢水平

**7、judge:**类型：数字

功能：记录用户在数据库中是否有个人信息的数据，有为1，没有为2

**8、\_id:**类型：字符串

功能：若用户在数据库中有个人信息的数据，则存放用户个人信息数据在数 据库中的数据识别码\_id，用于更新数据

**相关数据库的设计：**

**集合pri\_inf**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识符 | 类型 | 说明 |
| 数据id | \_id | string | 该条数据身份信息（主键） |
| openid | \_openid | string | 用户身份识别码 |
| 身高 | height | string | 用户当前身高 |
| 体重 | weight | string | 用户当前体重 |
| 年龄 | age | string | 用户年龄 |
| 性别 | sex | string | 用户性别 |
| 基本代谢水平 | metabolism | number | 用户基础代谢水平 |

上述局部变量中的height，weight，age，sex，metabolism会在用户确认后存入次数据库集合pri\_inf中，若用户没有个人信息数据（即judge=1），则添加一条数据；若用户有个人信息数据（即judge=2），则更新该条数据

3.3.6、算法和流程

用户在相应输入框输入个人身体信息，完成后点击界面上的确定按钮。若用户有非法输入（如未输入身高/体重/年龄或输入格式不对），则用户点击确定按钮后会弹出相应的弹窗警告；若用户输入全部符合要求，则点击确定按钮后数据传输至后台。

经过相应公式计算：女性基础代谢率=661+9.6\*体重(kg)+1.72\*身高(cm)-4.7\*年龄

     男性基础代谢率=67+13.73\*体重(kg)+5\*身高(cm)-6.9\*年龄

通过输出框显示计算得到的基本代谢水平，并通过弹窗询问用户是否将此数据更新至数据库中用于用户个人信息的管理，用户可以点击“是”和“否”两个按钮来执行操作。

若用户点击否，则该数据不更新至数据库。

若用户点击是，则在数据库中查询是否已有该用户条目（查询该用户的openid),若有，则更新该条目的数据；若没有，则添加条目。

3.3.7、函数说明

**1、onLoad()函数：**

所在文件：Calculate.js

功能：在界面加载时运行，具体功能有：

①在数据库中查询是否已有该用户条目，若有则获取该用户的基本代谢水 平。

②判断是否有从个人信息页面通过wx.reLaunch()传递过来的参数，若有， 则说明用户在更新个人信息，将用户原有的个人信息写入输入框中，便于 用户的修改。

参数：option。对象类型，存放了用户原有的身体信息，包括身高、体重、年 龄、性别。

**2、bindPickerChange()函数：**

所在文件：Calculate.js

功能：滑动选择框的实现，用于用户选择性别时滑动选择框中选定内容的改变。

参数：e。对象类型，里面的e.detail.value为当前选择内容所对应的下标。

**3、heightInput()函数：**

所在文件：Calculate.js

功能：记录用户输入的身高

参数：e。对象类型，里面的e.detail.value为当前用户在身高输入框中输入的内 容。

**4、weightInput()函数：**

所在文件：Calculate.js

功能：记录用户输入的体重

参数：e。对象类型，里面的e.detail.value为当前用户在体重输入框中输入的内 容。

**5、ageInput()函数：**

所在文件：Calculate.js

功能：记录用户输入的年龄

参数：e。对象类型，里面的e.detail.value为当前用户在年龄输入框中输入的内 容。

**6、calculate()函数：**

所在文件：Calculate.js

功能：①判断用户的输入是否合法，若不合法则给出相应的弹窗提示。

②通过以下公式计算用户的基础代谢水平：

女性基础代谢率=661+9.6\*体重(kg)+1.72\*身高(cm)-4.7\*年龄

  男性基础代谢率=67+13.73\*体重(kg)+5\*身高(cm)-6.9\*年龄

计算完成后给出相应的弹窗提示并询问用户是否更新个人数据。

③若用户点击是，则在数据库中查询是否已有该用户条目（查询该用户的openid),若有，则更新该条目的数据；若没有，则添加条目。

局部变量：height:类型：字符串 作用：用户输入的身高

weight:类型：字符串 作用：用户输入的体重

age:类型：字符串 作用：用户输入的年龄

index:类型：数字 作用：用户选择的性别所对应的下标

metabolism:类型：数字 作用：计算得到的用户基础代谢水平

judge:类型：数字 作用：判断数据库中是否已有该用户条目：

没有:judge=1;有：judge=2

\_id:类型：字符串 作用：用户的openid

**7、ToFoodSelect()函数：**

所在文件：Calculate.js

功能：界面跳转，调用API:wx.redirectTo，跳转至热量查询界面。

**8、ToPersonalData()函数：**

所在文件：Calculate.js

功能：界面跳转，调用API:wx.redirectTo，跳转至个人数据界面。

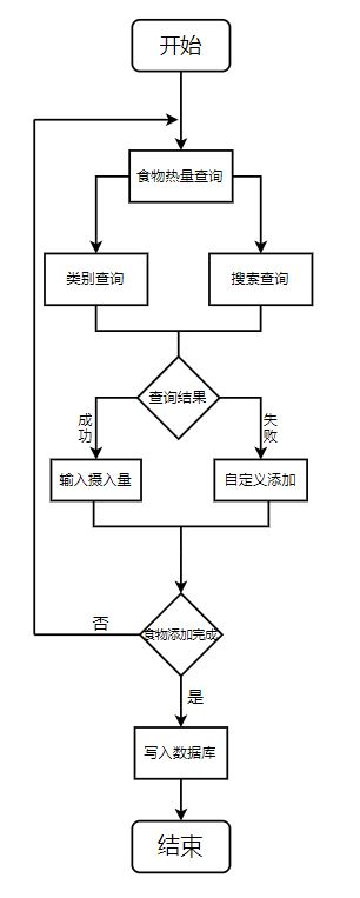
3.3.8 全局数据结构与该模块的关系

在onLoad和calculate函数中访问了全局变量openid，用于查询数据库中是否有用户的个人数据，如果没有则将局部变量judge赋值为1；如果有则将该条数据对应的\_id存入局部变量\_id中，并将局部变量judge赋值为2。

在onLoad函数中还访问了全局变量metabolism，用于存入推荐摄入热量RCI的最新数据和保证其他界面的正常显示。具体来说，只要用户进入代谢计算界面，就会查询用户在数据库中的基本代谢水平MET，若查询到，则将用户的RCI通过公式（MET \* 1.2）计算后存入全局变量metabolism，若没查询到则不操作；若用户更新个人数据，则将用户新的RCI通过公式（MET \* 1.2）计算后存入全局变量metabolism。

**3.4、模块2 热量查询模块**

3.4.1、设计图



3.4.2、功能描述

**功能1：通过多种方式查找食物，并获取食物热量信息：**用户可通过食物的分类进入不同的食物列表进行滑动寻找，也可以通过搜索框进行名称搜索（采用模糊搜索），对于小程序不包含的食物，可以手动添加摄入热量，严格管理

若通过搜索方式查找食物：

    输入的数据：食物名称

    通过模糊搜索的方式在数据库集合food中寻找对应的食物信息

得到的输出：数据库返回的食物信息数据

**功能2：添加食物摄入热量：**查询到指定的食物后，用户可通过输入每种食物摄入的 克数来添加摄入记录，对于数据库不包含的食物，还可以手动添加摄入热量，实现 严格管理记录。

选择查询到的食物添加摄入热量：

输入的数据：对应食物的摄入克数

    通过计算得到该食物摄入的热量，添加数据并更新总摄入热量

得到的输出：更新后的总摄入热量

自定义手动添加摄入热量；

输入的数据：对应食物的摄入热量

    添加数据，更新总摄入热量

得到的输出：更新后的总摄入热量

**功能3：管理食物摄入热量：**用户可以在摄入管理界面进行热量数据的具体细节查询（每种食物摄入的克数及其总热量），还支持分条删除数据，确认后可以提交数据进行历史记录。

3.4.3、输入数据

1、查询食物名称：数据类型：字符串

获取数据媒体：文本输入框

   格式：无

   有效性检验规则：判断是否输入数据（输入字符串是否为空）， 若为空，弹出相应的弹窗警告。

2、食物摄入克数：数据类型：数字

获取数据媒体：数字输入框

格式:XX.XX

有效性检验规则：1、判断是否输入数据（输入字符串是否为 空），若为空，弹出相应的弹窗警告。

2、正则表达式匹配，判断输入格式是否满足XX.XX，对应正则表达式为/^\d+(\.\d+)?$/。若不匹配，弹出 相应的弹窗警告。

3、自定义添加热量：数据类型：数字

   获取数据媒体：数字输入框

  格式:XX.XX

  有效性检验规则：1、判断是否输入数据（输入字符串是否为 空），若为空，弹出相应的弹窗警告。

2、正则表达式匹配，判断输入格式是否满足XX.XX，对应正则表达式为/^\d+(\.\d+)?$/。若不匹配，弹出 相应的弹窗警告。

以上输入数据的获取条件均为用户在应用热量查询模块时，通过键盘自愿主动输入

具体来说，查询食物名称通过查询界面中的输入框获取，食物摄入克数通过食物类别界面中的弹窗输入框中获取，自定义添加热量通过食物搜索界面、食物类别界面和热量查询界面中的弹窗输入框中获取

3.4.4、输出数据

1、已摄入热量：数据类型：数字

格式：XX.XX

表现形式：在底部信息栏显示相应数据（具体参照图例1）

2、推荐摄入热量：数据类型：数字

格式：XX.XX

表现形式：在底部信息栏显示相应数据（具体参照图例1）

descript

例1

3、已摄入食物列表：数据类型：数组

表现形式：如下图所示



4、食物搜索结果：数据类型：数组

表现形式：如下图所示



5、各类别的食物条目：数据类型：数组

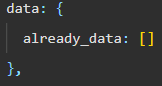
表现形式：如下图所示



3.4.5、数据设计

本模块的局部变量数据结构在不同的界面中有所不同

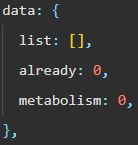
**摄入热量管理界面的局部变量**如下图所示



**already\_data:**类型：数组

功能：存放访问全局变量already\_data返回的用户已添加的食物数据

**食物类别界面的局部变量**如下图所示，食物类别界面共有6个，分别代表不同的食物类别，局部变量数据结构设计相同



**具体说明**：

**1、list:**类型：数组

功能：存放数据库集合food返回的对应类别的食物信息

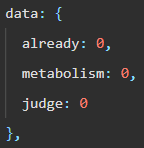
**2、already:**类型：数字

功能：存放访问全局变量already得到的用户的已摄入热量CT

**3、metabolism**:类型：字符串

功能：存放访问全局变量metabolism得到的用户的推荐摄入热量RCI

**热量查询界面的局部变量**如下图所示



**具体说明**：

**1、already:**类型：数字

功能：存放访问全局变量already得到的用户的已摄入热量CT

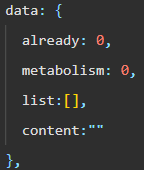
**2、metabolism**:类型：字符串

功能：存放访问全局变量metabolism得到的用户的推荐摄入热量RCI

**3、judge:**类型：数字

功能：存放历史数据判断位，辅助历史数据提交功能的实现，即查询用户在当前日期是否有历史数据，如果没有，则judge=1，若提交历史数据则添加数据；如果有，则judge=1，若提交历史数据则更新数据

**食物搜索界面的局部变量**如下图所示



**具体说明**：

**1、already:**类型：数字

功能：存放访问全局变量already得到的用户的已摄入热量CT

**2、metabolism**:类型：字符串

功能：存放访问全局变量metabolism得到的用户的推荐摄入热量RCI

**3、list:**类型：数组

功能：存放数据库集合food返回的符合模糊搜索结果的食物信息

**4、content:**类型：字符串

功能：存放用户搜索输入的食物名称

**相关数据库的设计：**

**集合food**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识符 | 类型 | 说明 |
| 数据id | \_id | string | 该条数据身份信息（主键） |
| 名称 | name | string | 食物名称 |
| 单位热量 | cal | string | 食物单位热量（每100g） |
| 类别 | class | string | 食物类别 |
| 图片 | img | string | 食物图片链接（云存储链接） |

上述局部变量中的list会存入数据库集合food返回的食物信息，并配合前端设计开发显示食物信息

**集合history**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识符 | 类型 | 说明 |
| 数据id | \_id | string | 该条数据身份信息（主键） |
| openid | \_openid | string | 用户身份识别码 |
| 摄入热量 | cal | string | 用户历史摄入热量 |
| 日期 | date | string | 用户历史热量摄入对应的日期 |

上述局部变量中的already会在用户点击完成按钮并确认后存入数据库集合history中（存储的键为cal，同时存入全局变量date），若用户没有对应日期的历史数据（即judge=1），则添加一条数据；若用户有对应日期的历史数据（即judge=2），则更新该条数据

给出本程序中的局部数据结构说明，包括数据结构名称，功能说明，具体数据结构说明（定义、注释设计、取值）等。相关数据库表，数据存储设计（具体说明需要以文件方式保存的数据文件名、数据存储格式、数据项及属性等。）

3.4.6、算法和流程

热量查询模块的所有界面（除了摄入热量管理界面之外）在底部（或导航栏上方）均会显示已摄入热量CT和推荐摄入热量RCI的数据，并且还有descript按钮支持跳转到摄入热量管理界面，“完成”按钮支持提交摄入热量数据到数据库。

descript

并且在界面顶部均有输入搜索框和“手动添加摄入热量”按钮，分别用于跳转到食物搜索界面和打开自定义摄入热量弹窗

descript

用户首先会进入热量查询界面，界面中除了上述的元素，还有6个不同的食物类别按钮

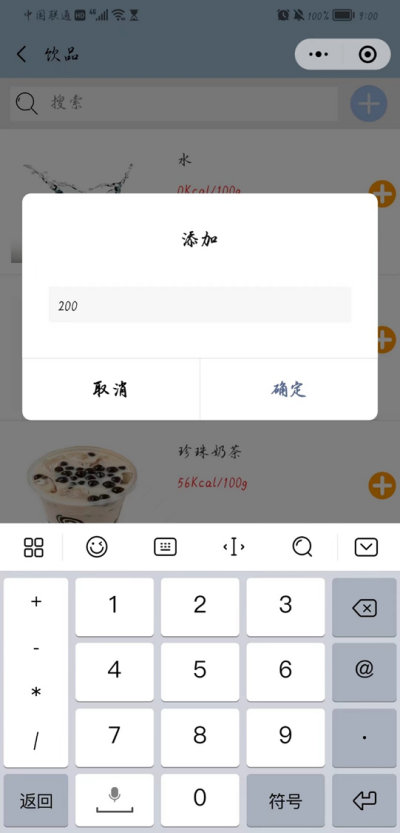


用户可以点击对应的类别按钮进入食物类别界面

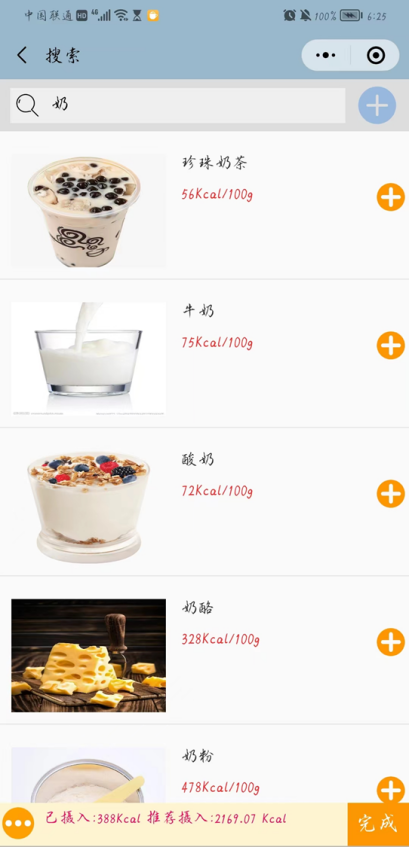
点击食物类别进入不同的食物选择界面后（最左图为主食界面），可以上下滑动进行食物热量的查找以及食物的选择



点击对应食物右侧的加号后就可以输入摄入的克数了，输入完成后再次点击确定则添加该食物摄入的热量数据（新增摄入热量=单位热量\*克数/100），添加后即可在界面下方的“已摄入”模块看到热量变化（CT=CT+新增摄入热量）



当通过界面顶部的搜索栏进行食物搜索时，直接在搜索栏中输入对应的食物名称，点击键盘上的搜索（PC端按下回车）即可完成搜索。此处的搜索采用模糊搜索方式，即只要名称中存在搜索的字段就会命中，相对于精准搜索，模糊搜索方式不会因为食物名称的不同叫法而搜索失败，更加的人性化



若数据库中不存在用户搜索的食物，系统会提示搜索结果为空，并引导用户点击搜索框右侧的“+”手动添加数据



用户可在自行查询热量摄入之后手动添加该食物的热量摄入数据，点击确定则添加该自定义摄入热量数据，添加后即可在界面下方的“已摄入”模块看到热量变化（CT=CT+手动输入摄入热量）



点击热量查询页面左下方的descript按钮，即可跳转至已添加的食物——摄入热量管理界面，此界面显示了每种食物摄入的克数和总热量，并支持删除数据进行更改（CT=CT-被删除数据的摄入热量）



为了防止误操作，在删除数据前系统会弹窗提示进行确认



完成当日食物的选取，也就是热量摄入的管理后，点击热量查询界面右下方的“完成”按钮，系统会弹窗进行确认，点击确认后当日热量热量的数据便会添加至后端云数据库，若要进行数据的更改，只需要重新选择食物并添加删除数据，完成后再次点击“完成”和“确认”，即可更新当日的热量摄入数据



3.4.7、函数说明

**一、热量查询界面(foodSelect文件夹)：**

**1、onShow()函数：**

所在文件：foodSelect.js

功能：在界面显示时执行，调用setData方法，将推荐摄入热量RCI和基本代谢水平MET渲染至视图层。

局部变量：metabolism 类型：数字 作用：存放基本代谢水平MET

already 类型：数字 作用：存放推荐摄入热量RCI

**2、ToCalculate()函数：**

所在文件：foodSelect.js

功能：界面跳转，调用API:wx.redirectTo，跳转至代谢计算界面。

**3、ToPersonalData()函数：**

所在文件：foodSelect.js

功能：界面跳转，调用API:wx.redirectTo，跳转至个人数据界面。

**4、ToRice()函数：**

所在文件：foodSelect.js

功能：界面跳转，调用API:wx.navigateTo，跳转至主食类别界面。

**5、ToVegetable()函数：**

所在文件：foodSelect.js

功能：界面跳转，调用API:wx.navigateTo，跳转至蔬果类别界面。

**6、ToMeat()函数：**

所在文件：foodSelect.js

功能：界面跳转，调用API:wx.navigateTo，跳转至肉类界面。

**7、ToDrink()函数：**

所在文件：foodSelect.js

功能：界面跳转，调用API:wx.navigateTo，跳转至饮品类别界面。

**8、ToNut()函数：**

所在文件：foodSelect.js

功能：界面跳转，调用API:wx.navigateTo，跳转至坚果类别界面。

**9、ToOthers()函数：**

所在文件：foodSelect.js

功能：界面跳转，调用API:wx.navigateTo，跳转至其它类别界面。

**10、ToSearch()函数：**

所在文件：foodSelect.js

功能：界面跳转，调用API:wx.navigateTo，跳转至食物搜索界面。

**11、Add()函数：**

所在文件：foodSelect.js

全局变量：already 类型：数字 作用：存放推荐摄入热量CT

already\_data 类型：数组 作用：存放用户已添加的食物数据和手动 输入的热量数据

功能：用户手动添加摄入热量。判断用户输入的热量是否合法，不合法则弹出相 应的弹窗警告；合法则更新变量already和already\_data的值

**12、upload()函数：**

所在文件：foodSelect.js

功能：当用户完成食物的选取后，点击完成按钮。弹出弹窗提示“您确定已完成食物的选取？”防止误触产生。若用户点击确认。则向数据库中查询是否有用户当天的历史数据。若有，则在该条目的基础上进行修改；若没有，则增添该日记录条目。

**13、ToAlready()函数：**

所在文件：foodSelect.js

功能：界面跳转，调用API:wx.navigateTo，跳转至摄入热量管理界面。

**二、食物类别界面(food文件夹下的6个文件，分别对应6个不同类别，以下以饮品类别drink文件夹为例)：**

1. **getList()函数：**

所在文件：drink.js

功能：在数据库中查询食物类别为“饮品”的食物条目，并装入一个数组list中。

局部变量：list 类型：数组 作用：存放饮品类别的食物数据

**2、onLoad()函数：**

所在文件：drink.js

功能：在界面首次加载时执行，有两个功能：

1、调用getList()函数，从数据库中查找该类别的食物并返回一个数组list。

2、调用setData方法，将推荐摄入热量RCI和基本代谢水平MET渲染至视图层。

局部变量：metabolism 类型：数字 作用：存放基本代谢水平MET

already 类型：数字 作用：存放推荐摄入热量RCI

list 类型：数组 作用：存放饮品类别的食物数据

**3、onShow()函数：**

所在文件：drink.js

功能：在界面显示时执行，调用setData方法，将推荐摄入热量RCI和基本代谢水平MET以及list渲染至视图层。

局部变量：metabolism 类型：数字 作用：存放基本代谢水平MET

already 类型：数字 作用：存放推荐摄入热量RCI

list 类型：数组 作用：存放饮品类别的食物数据

**4、ToSearch()函数：**

所在文件：drink.js

功能：界面跳转，调用API:wx.navigateTo，跳转至食物搜索界面。

**5、ToAlready()函数：**

所在文件：drink.js

功能：界面跳转，调用API:wx.navigateTo，跳转至摄入热量管理界面。

**6、Add()函数：**

所在文件：drink.js

全局变量：already 类型：数字 作用：存放推荐摄入热量CT

already\_data 类型：数组 作用：存放用户已添加的食物数据和手动 输入的热量数据

功能：用户手动添加摄入热量。判断用户输入的热量是否合法，不合法则弹出相 应的弹窗警告；合法则更新变量already和already\_data的值

**7、upload()函数：**

所在文件：drink.js

功能：当用户完成食物的选取后，点击完成按钮。弹出弹窗提示“您确定已完成食物的选取？”防止误触产生。若用户点击确认。则向数据库中查询是否有用户当天的历史数据。若有，则在该条目的基础上进行修改；若没有，则增添该日记录条目。

**8、addfood()函数：**

所在文件：drink.js

全局变量：already 类型：数字 作用：存放推荐摄入热量CT

already\_data 类型：数组 作用：存放用户已添加的食物数据和手动 输入的热量数据

功能：用户点击食物条目右侧的“+”号添加数据，在弹出窗口的输入框中输出摄入克数，采用正则表达式匹配的方式检验用户输入格式的合法性。若不合法，则弹出相应的弹窗提示；若合法则将食物摄入量换算成摄入热量，之后更新变量already和already\_data的值。

**三、食物搜索界面(Search文件夹)：**

**1、onLoad()函数：**

所在文件：Search.js

功能：在界面显示时执行，调用setData方法，将推荐摄入热量RCI和基本代谢水平MET渲染至视图层。

局部变量：metabolism 类型：数字 作用：存放基本代谢水平MET

already 类型：数字 作用：存放推荐摄入热量RCI

**2、ToAlready()函数：**

所在文件：Search.js

功能：界面跳转，调用API:wx.navigateTo，跳转至摄入热量管理界面。

**3、Add()函数：**

所在文件：Search.js

全局变量：already 类型：数字 作用：存放推荐摄入热量CT

already\_data 类型：数组 作用：存放用户已添加的食物数据和手动 输入的热量数据

功能：用户手动添加摄入热量。判断用户输入的热量是否合法，不合法则弹出相 应的弹窗警告；合法则更新变量already和already\_data的值

**4、upload()函数：**

所在文件：Search.js

功能：当用户完成食物的选取后，点击完成按钮。弹出弹窗提示“您确定已完成食物的选取？”防止误触产生。若用户点击确认。则向数据库中查询是否有用户当天的历史数据。若有，则在该条目的基础上进行修改；若没有，则增添该日记录条目。

**5、addfood()函数：**

所在文件：Search.js

全局变量：already 类型：数字 作用：存放推荐摄入热量CT

already\_data 类型：数组 作用：存放用户已添加的食物数据和手动 输入的热量数据

功能：用户点击食物条目右侧的“+”号添加数据，在弹出窗口的输入框中输出摄入克数，采用正则表达式匹配的方式检验用户输入格式的合法性。若不合法，则弹出相应的弹窗提示；若合法则将食物摄入量换算成摄入热量，之后更新变量already和already\_data的值。

**6、Search()函数：**

所在文件：Search.js

参数：res 类型：对象 作用：res.detail.value中存放了用户在搜索框中输入的字符串

局部变量：list 类型：数组 作用：存放与搜索名称相匹配的食物

功能：采用模糊搜索（正则表达式匹配）的方式在数据库中搜索用户输入的食物名称，将搜索名称相匹配的食物存入数组list。

**四、摄入热量管理界面(already文件夹)**

**1、onShow()函数：**

所在文件：already.js

全局变量：already\_data类型：数组 作用：存放用户已添加的食物数据和手动输 入的热量数据

功能：在界面显示时执行，调用setData方法，将全局变量already\_data渲染至视图层。

1. **Del()函数：**

所在文件：already.js

参数：e 类型：对象 作用：e.currentTarget.dataset.index中指定了用户 点击删除的食物条目的下标

全局变量： already 类型：数字 作用：存放推荐摄入热量CT

already\_data 类型：数组 作用：存放用户已添加的食物数据和手动输入的热量数据

功能：用户点击食物条目右侧的删除按钮后，弹窗提示用户“是否要删除该条数据”。若用户点击确认，则从already\_data数组中删除对应下标的数据，并从already中减去该条数据对应的热量，最后弹窗提示用户删除成功。

3.4.8 全局数据结构与该模块的关系

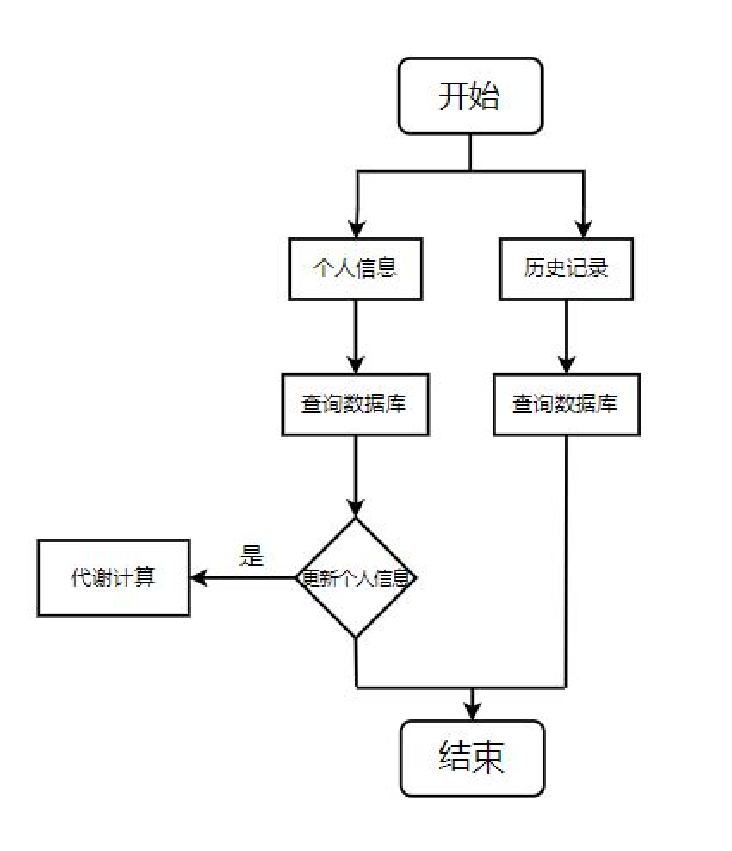
在food文件夹中的6个js文件（即6个食物类别界面）、热量查询界面foodSelect.js和食物查询界面Search.js中访问了全局变量already和metabolism，分别用于获取用户的已摄入热量CT和推荐摄入热量RCI，使前端界面正常显示这两个数据

在摄入热量管理界面already.js中还访问了全局变量already\_data，用于获取用户已添加的食物数据，并通过前端将已摄入食物的种类、热量和克数显示在界面上

**3.5、模块3 个人数据模块**

详细描述各功能模块的功能、数据结构、具体算法和流程等。

3.5.1、设计图



3.5.2、功能描述

功能1：历史数据查询：系统会以天为单位，存储用户提交的每日摄入总热量的数据， 并支持当天内进行数据的修改更新，用户可通过历史数据模块进行查询

功能2：个人信息管理：系统会将用户提交的基本身体信息显示在个人数据界面，还支 持通过更新数据按钮跳转到代谢计算界面，并将用户原本的数据写入，方便修改更新

3.5.3、输入数据：无

3.5.4、输出数据

1、用户身体数据：数据类型：对象，内容包括身高、体重、年龄、性别、基础代谢

表现形式：如下图所示



2、用户历史数据：数据类型：对象，内容包括日期和对应摄入热量

表现形式：如下图所示



3、调试信息：数据类型：字符串

表现形式：在控制台输出相应的提示信息，如“用户基本信息请求成功”等，用于开发者的调试。

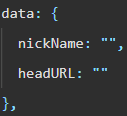
4、用户账户信息：数据类型：对象，内容包括微信头像url、微信昵称

表现形式：如下图所示



3.5.5、数据设计

本模块的局部变量数据结构在不同的界面中有所不同

**个人数据界面的局部变量**如下图所示

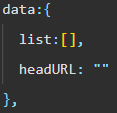
**具体说明：**

**nickName:**类型：字符串

功能：存放全局变量返回的用户昵称nickName，使前端界面能够正常显示 用户昵称

**headURL:**类型：字符串

功能：存放全局变量返回的用户头像URL headURL数据，使前端界面能够正 常显示用户头像

**个人基本信息界面的局部变量**如下图所示

**具体说明：**

**list:**类型：数组

功能：存放数据库集合pri\_inf返回的用户个人基本身体信息

**headURL:**类型：字符串

功能：存放全局变量返回的用户头像URL headURL数据，使前端界面能够正 常显示用户头像

**历史数据界面的局部变量**如下图所示

**具体说明：**

**list:**类型：数组

功能：存放数据库集合history返回的用户历史信息

**相关数据库的设计：**

**集合pri\_inf**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识符 | 类型 | 说明 |
| 数据id | \_id | string | 该条数据身份信息 |
| openid | \_openid | string | 用户身份识别码 |
| 身高 | height | string | 用户当前身高 |
| 体重 | weight | string | 用户当前体重 |
| 年龄 | age | string | 用户年龄 |
| 性别 | sex | string | 用户性别 |
| 基本代谢水平 | metabolism | number | 用户基础代谢水平 |

上述个人基本信息界面局部变量中的list会从数据库集合pri\_inf中获取用户的个人信息数据，并将height,weight,age,sex,metabolism显示在前端页面上

**集合history**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识符 | 类型 | 说明 |
| 数据id | \_id | string | 该条数据身份信息 |
| openid | \_openid | string | 用户身份识别码 |
| 摄入热量 | cal | string | 用户历史摄入热量 |
| 日期 | date | string | 用户历史热量摄入对应的日期 |

上述历史数据界面局部变量中的list会从数据库集合history中获取用户的历史数据，并将cal和date显示在前端页面上

3.3.6、算法和流程

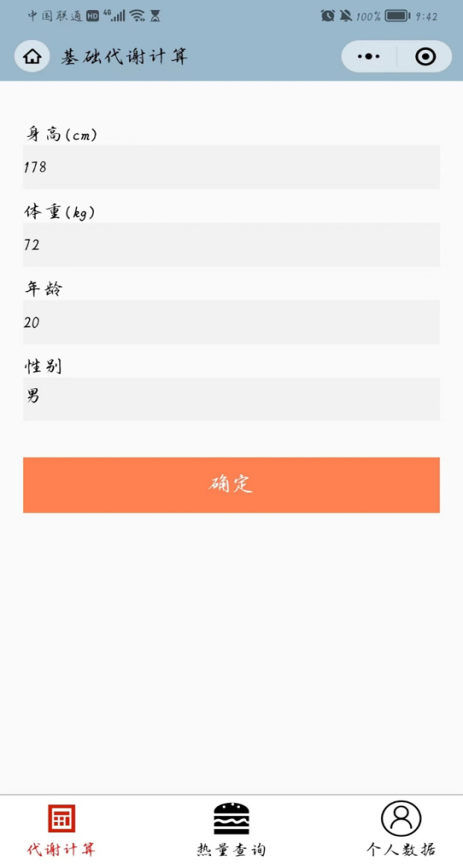
通过点击界面下方的导航栏可以跳转到个人数据界面，该界面上方显示了登录微信号的头像和昵称，界面中部的两个按钮选项可以分别导航至个人信息界面和历史记录界面



点击跳转到个人信息界面，该界面上方显示用户微信头像，中部显示该用户的个人基本信息，包括身高、体重、年龄、性别和经计算得到的基本代谢水平



若发现信息有误或者身体信息发生变更时，可以通过点击“更新个人信息”按钮进行数据的更新，点击后系统会跳转到代谢计算界面并将用户原本的信息自动填入输入框中，方便用户更新数据信息



3.5.7、函数说明

1. **个人数据界面（PersonalData文件夹):**

**1、onLoad()函数：**

所在文件：PersonalData.js

功能：在界面首次加载时运行，调用setData方法，将用户微信头像和微信昵称渲染至视图层。

局部变量：nickname 类型：字符串 作用：用户的微信昵称

headURL 类型：字符串 作用：用户微信头像的url

**2、ToFoodSelect()函数：**

所在文件：PersonalData.js

功能：界面跳转，调用API:wx.redirectTo，跳转至热量查询界面。

**3、ToCalculate()函数：**

所在文件：PersonalData.js

功能：界面跳转，调用API:wx.redirectTo，跳转至代谢计算界面。

**4、ToInformation()函数：**

所在文件：PersonalData.js

功能：界面跳转，调用API:wx.navigateTo，跳转至个人信息界面。

**5、ToCalculate()函数：**

所在文件：PersonalData.js

功能：界面跳转，调用API:wx.navigateTo，跳转至历史记录界面。

1. **个人信息界面（Information文件夹）**

**1、onLoad()函数：**

所在文件：Information.js

功能：在界面首次加载时运行，功能包括：

1、在数据库中查询当前用户的基本信息，存入数组list中。

2、将用户微信头像和用户的基本信息list渲染至视图层。

局部变量：list 类型：数组 作用：存放用户的基本信息，包括身高、体重、年 龄、性别、基础代谢水平

headURL 类型：字符串 作用：用户微信头像的url

**2、ToCalculate()函数：**

所在文件：Information.js

功能：在用户点击“更新个人信息”按钮后进行界面跳转。调用API:wx.reLaunch，跳转至代谢计算界面，并携带参数height、weight、age、sex。

局部变量：height 类型：数字 作用：用户身高

weight 类型：数字 作用：用户体重

age 类型：数字 作用：用户年龄

sex 类型：字符串 作用：用户性别

1. **历史记录界面（History文件夹）**

**onLoad()函数：**

所在文件：History.js

功能：在界面首次加载时运行，功能包括：

1. 在数据库中查询当前用户的历史数据并存入数组list中。
2. 用setData方法将用户的历史数据list渲染至视图层。

具体说明模块中的各个函数，包括函数名称及其所在文件，功能，格式，参数，全局变量，局部变量，返回值，算法说明，使用约束等。

3.5.8 全局数据结构与该模块的关系

在个人数据界面PersonalData.js的onLoad函数中访问了全局变量userInfo对象中userInfo.nickName的值和userInfo.avatarUrl的值，分别用于获取用户的昵称和头像URL

在个人基本信息界面Information.js的onLoad函数中访问了全局变量userInfo对象中userInfo.avatarUrl的值，用于获取用户的头像URL，还访问了全局变量openid，作为用户的唯一身份识别码，用于查询数据库集合pri\_inf中用户的个人基本信息数据，并通过前端显示在界面上

在历史数据界面History.js的onLoad函数中访问了全局变量openid，作为用户的唯一身份识别码，用于查询数据库集合history中用户的所有历史数据，并通过前端显示在界面上

# 4、 接口设**计**

**4.1、 用户接口**

名称：wx.getUserProfile

作用：请求用户对小程序的授权，用户授权登录，记录登录状态，并提供一些账户信息给本软件，说明将向用户提供的接口

输入信息：无

输出信息：无

异常处理：若用户拒绝授权或获取用户个人信息失败，则小程序无法正常运行，自动关闭退出

**4.2、 外部接口**

**控制方式：**程序方式

**软件接口：**通过Node.js与数据库之间进行交互

**硬件接口：**鼠标、键盘等

**接口关系：**小程序前端界面与后端云数据库的接口

1、名称：GetHistory

作用：从数据库中调出相应用户的历史数据记录

输入信息：用户账户的openid

输出信息：对应微信用户的历史数据，包括历史日期和对应摄入的总热量。存放在一个数组中

异常处理：若用户为新用户，则返回无历史记录的提示

2、 名称：SetData

作用：更新用户的个人数据

输入信息：用户输入的身高、体重、性别、年龄、基础代谢水平

输出信息：调试信息，如“寻找数据请求成功”、“添加成功”等

异常处理：若写入数据库失败，则在控制台输出相应的错误信息提示，如“更新失败”

3、名称：SetCalorie

作用：更新已摄入食物及总热量的数据并将其与当前日期存入数据库中

输入信息：已摄入食物种类以及对应克数或手动输入的食物总热量，当前手机的日期

输出信息：已摄入的总热量，调试信息，如“寻找数据请求成功”、“添加成功”等

异常处理：若写入数据库失败，则在控制台输出相应的错误信息提示，如“更新失败”

4、名称：GetData

作用：从数据库中调出相应用户的个人基本信息

输入信息：用户账户的openid

输出信息：对应微信用户的个人基本信息，包括身高、体重、性别、年龄、基础代谢水平

。存放在一个数组中

异常处理：若用户为新用户，则返回无个人基本信息记录的提示

**4.3、 内部接口**

4.3.1、 接口说明

1、名称：ToCalculate

作用：调用路由API:wx.redirectTo,从当前模块（代谢计算模块、热量查询模块、个人数据模块）跳转到代谢计算模块

异常处理：调用该接口时调用API:wx.showLoading,显示加载中弹窗；若界面跳转成功，弹窗消失；若界面跳转失败，弹出相应的提示弹窗。

2、名称：ToPersonalData

作用：调用路由API:wx.redirectTo,从当前模块（代谢计算模块、热量查询模块、个人数据模块）跳转到个人数据模块

异常处理：调用该接口时调用API:wx.showLoading,显示加载中弹窗；若界面跳转成功，弹窗消失；若界面跳转失败，弹出相应的提示弹窗。

3、名称：ToFoodSelect

作用：调用路由API:wx.redirectTo,从当前模块（代谢计算模块、热量查询模块、个人数据模块）跳转到热量查询模块

异常处理：调用该接口时调用API:wx.showLoading,显示加载中弹窗；若界面跳转成功，弹窗消失；若界面跳转失败，弹出相应的提示弹窗。

4、名称：ToInformation

作用：调用路由API:wx.navigateTo,从个人数据模块跳转至个人信息模块

异常处理：调用该接口时调用API:wx.showLoading,显示加载中弹窗；若界面跳转成功，弹窗消失；若界面跳转失败，弹出相应的提示弹窗。

5、名称：ToHistory

作用：调用路由API:wx.navigateTo,从个人数据模块跳转到历史数据模块

异常处理：调用该接口时调用API:wx.showLoading,显示加载中弹窗；若界面跳转成功，弹窗消失；若界面跳转失败，弹出相应的提示弹窗。

6、名称：ToFood

作用：调用路由API:wx.navigateTo,从热量查询模块跳转到食物类别模块（包括主食、蔬果、肉类、坚果、饮品和其它6个界面）

异常处理：调用该接口时调用API:wx.showLoading,显示加载中弹窗；若界面跳转成功，弹窗消失；若界面跳转失败，弹出相应的提示弹窗。

7、名称：ToSearch

作用：调用路由API:wx.navigateTo,从当前模块（热量查询模块、食物类别模块）跳转到食物搜索模块

异常处理：调用该接口时调用API:wx.showLoading,显示加载中弹窗；若界面跳转成功，弹窗消失；若界面跳转失败，弹出相应的提示弹窗。

8、名称：ToUpdate

作用：调用路由API:wx.redirectTo,从个人基本信息模块跳转到代谢计算模块

输入信息：用户原本的个人基本信息（身高、体重、年龄、性别）

异常处理：调用该接口时调用API:wx.showLoading,显示加载中弹窗；若界面跳转成功，弹窗消失；若界面跳转失败，弹出相应的提示弹窗。

9、名称：ToAlready

作用：调用路由API:wx.navigateTo,从当前模块跳转到摄入热量管理模块

异常处理：调用该接口时调用API:wx.showLoading,显示加载中弹窗；若界面跳转成功，弹窗消失；若界面跳转失败，弹出相应的提示弹窗。

4.3.2、 调用方式

以上均为路由接口，在需要调用的页面的wxml中用bindtap将接口与页面元素进行绑定即可。例：<view class="guide-item" bindtap="ToPersonalData">

若要向跳转页面传递特定参数，则需要在调用页面的.js文件中对该接口函数进行重构。具体方法为：在需要跳转的应用内页面路径后添加参数。参数与路径之间使用?分隔，参数键与参数值用=相连，不同参数用&分隔；如 'path?key=value&key2=value2'

例：

未重构的原生接口，实现跳转至url="test"的界面，没有要传递的参数

ToCalculate: function (e) {  
 wx.reLaunch({  
 url: 'test'  
 })  
}

重构后，实现跳转至url="test"的界面，传递参数id=1,age=2

ToCalculate: function (e) {  
 wx.reLaunch({  
 url: 'test?id=1&age=2'  
 })   
}

# 5、数据库设计

数据库系统和数据库运行环境：均由微信开发者工具云开发平台提供

本小程序后端云数据库共有三个集合，分别为food,history,pri\_inf

**集合food：功能：用于存放食物信息数据**

**数据权限设置：所有用户可读，仅创建者可读写**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识符 | 类型 | 说明 |
| 数据id | \_id | string | 该条数据身份信息（主键） |
| 名称 | name | string | 食物名称 |
| 单位热量 | cal | string | 食物单位热量（每100g） |
| 类别 | class | string | 食物类别 |
| 图片 | img | string | 食物图片链接（云存储链接） |

**集合history：功能：用于存放所有用户的历史数据**

**数据权限设置：所有用户可读写**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识符 | 类型 | 说明 |
| 数据id | \_id | string | 该条数据身份信息（主键） |
| openid | \_openid | string | 用户身份识别码 |
| 摄入热量 | cal | string | 用户历史摄入热量 |
| 日期 | date | string | 用户历史热量摄入对应的日期 |

**集合pri\_inf：功能：用于存放所有用户的个人基本信息**

**数据权限设置：所有用户可读写**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识符 | 类型 | 说明 |
| 数据id | \_id | string | 该条数据身份信息（主键） |
| openid | \_openid | string | 用户身份识别码 |
| 身高 | height | string | 用户当前身高 |
| 体重 | weight | string | 用户当前体重 |
| 年龄 | age | string | 用户年龄 |
| 性别 | sex | string | 用户性别 |
| 基本代谢水平 | metabolism | number | 用户基础代谢水平 |

存储要求：要求存储无差错，时间长且便于快速索引

访问方法：在对应的数据表中以\_id或\_openid或name或class为主键进行索引

存取物理关系：

存储区域：根据常用部分与不常用部分可以将系统数据存放在两个磁盘上面，系统中的用户历史数据记录经常需要增加、删除、修改以及查询，故data和calorie应该存放在一起，而个人基本信息只是在用户身体数据发生大幅改变时才会进行修改，或者有新用户注册时才需要增加，所以\_id、\_openid、height、weight、age、sex、metabolism应该放在另一个磁盘上面

设备：可正常运行微信客户端的硬件设备

索引：在对应的数据表中以\_id或\_openid或name或class为主键进行索引

保密条件：除开发者外，用户的个人基本信息和历史数据严格保密，无法通过小程序进行窃取

# 6、系统出错处理

**6.1、 出错信息**

|  |  |  |
| --- | --- | --- |
| 可能出现的错误与故障 | 系统输出信息的形式及含义 | 处理方式 |
| 数据库被破坏 | 在控制台和调试日志中输出“数据库异常” | 自动采用另一主机工作，使用每周更新的备份文件恢复数据库，并优化代码阻止入侵，最后联系微信小程序平台寻求帮助 |
| 同时上线用户过多，导致无法查询 | 在控制台和调试日志中输出“数据库拥塞” | 控制数据库每秒的访问次数或暂时关闭数据库的访问及修改功能 |

**6.2、 补救措施**

（1）为防止数据库可能被破坏，故采用双主机形式，当一个坏掉的时候，自动采用另外一个主机进行工作；并定期进行备份，比如一周

（2）为同时防止用户过多，应设置系统一次运行时间为30分钟，并且人工规定能同时上线的人数

# 7、其他设计

**7.1 运行设计：**

7.1.1 运行模块组合

通过主应用程序组合，模块均可独立操作存储文件

7.1.2 运行控制

用户只需要通过鼠标、键盘或者触控就可以实现信息输入、食物查找等一切操作

7.1.3 运行时间

基本代谢水平和热量计算时间<0.1s，个人历史信息查询时间<2s，个人基本信息查询时间<2s，食物类别界面渲染时间<3s，食物搜索结果显示时间<3s

**7.2 安全保密设计：**

系统的系统用户管理保证了只有授权的用户才能进入系统进行数据操作，而且对一些重要数据，系统设置为只有更高权限的人员方可读取或是操作，系统安全保密性较高

**7.3 维护设计：**

在一些与数据库连接的程序中，由于经常出现连接不成功的情况，故应该进行标记，当出现类似情况的时候方便进行维护